IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

SRI INTERNATIONAL, INC., a California Corporation, Plaintiff and Counterclaim-Defendant, v. INTERNET SECURITY SYSTEMS, INC., a Delaware corporation, INTERNET SECURITY SYSTEMS, INC., a Georgia corporation, and SYMANTEC CORPORATION, a Delaware corporation, Defendants and Counterclaim-Plaintiffs.	C A. No. 04-1199 (SLR) **PUBLIC VERSION**

DECLARATION OF DR. GEORGE KESIDIS IN SUPPORT OF SRI INTERNATIONAL, INC.'S RESPONSES TO:

(1) DEFENDANTS' JOINT MOTION TO LIMIT THE TESTIMONY OF EXPERT, DR. GEORGE KESIDIS, UNDER FEDERAL RULE OF EVIDENCE 702;

(2) DEFENDANTS' JOINT MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. §§ 102 AND 103;

(3) SYMANTEC'S MOTION FOR SUMMARY JUDGMENT OF NONINFRINGEMENT; AND

(4) ISS'S MOTION FOR SUMMARY JUDGMENT THAT THE ASSERTED CLAIMS OF THE SRI PATENTS-IN-SUIT ARE NOT INFRINGED OR, IN THE ALTERNATIVE, ARE INVALID

I, George Kesidis, declare as follows:

1. I am employed by the Pennsylvania State University. My work address is CSE Dept., 338J IST Building, University Park, PA, 16802, USA. I have also been retained by SRI to render expert opinions and testimony in this matter. I have personal knowledge of the matters stated in this declaration and would testify truthfully to them if called upon to do so.

Expert Qualifications and Knowledge of the Relevant Art

2. I earned a bachelor's degree in electrical engineering in 1988, a master's degree in electrical engineering and computer science ("EECS") in 1990, and a doctorate in EECS from the University of California at Berkeley in 1992.

3. I have conducted continuous research into communications and computer networking since beginning my graduate studies at Berkeley in 1990.

4. I am employed as a tenured Associate Professor (and will become a full Professor on July 1, 2006) in the Electrical Engineering and Computer Science & Engineering Departments of the Pennsylvania State University.

5. Since receiving my Ph.D. in 1992, I have taught courses and written articles on network traffic engineering including traffic modeling, statistical analysis, packet scheduling, and router control methods.

6. Between 1992 and the present date, I obtained numerous grants and consulting contracts in the field of network traffic engineering, a sibling field of cyber security. A significant portion of my research was in the sector of network traffic measurements and associated statistics, an area of overlap between the network traffic engineering and cyber security fields.

7. I began conducting research in the specific field of cyber security in 2000 and by 2001 my collaborators and I had drafted research proposals and papers for submission in this field.

8. When I first entered the cyber security field, I spent several months during the summer and fall of 2000 surveying the literature in order to familiarize myself with earlier work. I also attended security-related talks at several conferences. My review focused mostly on articles and conference proceedings from the previous two years. The review was initially a broad survey of the cyber security field, but I then focused more specifically on areas such as attribution and traceback – research having to do with identifying the source of a suspected attack – and thus read more extensively and further

back in time on those and closely-related subjects. I specifically recall reviewing articles about GrIDS in particular during that timeframe.

9. In 2002, I was awarded a $53,000 grant by Cisco Systems to explore how rapid traceback could be used to improve defenses against diffuse worm and more focused Distributed Denial of Service (DDoS) attacks.

10. In 2003, I began working on a high-profile, multiple-university project on the testing of cyber security defenses. This project, known as EMIST – together with its sister testbed project, DETER – has so far received $11 million in funding from the Department of Homeland Security and the National Science Foundation.

11. In addition to co-investigators of EMIST/DETER (including Philip Porras, one of the named inventors on the patents-in-suit, Felix Wu, and Karl Levitt), I have consulted with numerous other individuals widely known for their expertise in cyber security at conferences and NSF panels in an effort to ascertain the critical problems in the field, previous approaches proposed to address them, and currently promising directions within the field.

12. I have written two books, published more than twenty-five journal articles, and authored over 50 conference papers in the network traffic engineering field, more than one-quarter of which fall specifically within the cyber security discipline.

13. I have also been invited to present more than a dozen talks about this research, including at an internal Cisco Systems security summit, at the Department of Homeland Security, and at universities across North America. In June 2005, I delivered a lecture on the testing of Internet security systems in Tokyo at the 2nd U.S.-Japan Experts Workshop on Critical Information Infrastructure Protection.

14. In addition, I serve as a technical program committee co-chair of the Institute of Electrical and Electronics Engineers ("IEEE") INFOCOM 2007, a major comprehensive networking conference. I have held similar positions at numerous other IEEE conferences, including those that are specific to cyber security (like IEEE

NPSec'05 and ACM SIGCOMM LSAD'06), and I currently serve as a senior member of the IEEE.

15. Based on my review of the literature within that time frame, both at the time I entered the field and over the time I have been actively conducting research, I believe there was no material change in the level of ordinary skill in the field of cyber security between 1998 and 2000-2001. A person of this ordinary level of skill would possess either a bachelor's degree in Computer Science, Electrical Engineering, or Computer Engineering and 5-7 years experience in enterprise-deployed cyber security (or a master's degree in CS, EE, or CE and 3-5 years experience in same). Such a person would also be familiar with the basic, existing techniques of cyber security and their performance and limitations, as well as deployment and maintenance issues of cyber security products.

The Patents-in-suit and "Statistical Detection Methods"

16. I understand that Defendants have asserted that I "admitted" that use of fixed thresholds as part of an intrusion detection analysis always constitutes a "signature-based" method. My deposition testimony, wherein I acknowledge that generating a report of suspicious activity based on a fixed threshold count of failed login attempts can be characterized as a "signature" is in no way an "admission" that *any use* of a fixed threshold is a "signature," and therefore not "statistical" analysis. A threshold can certainly be used as part of a statistical detection algorithm. For example, the threshold itself could be statistically derived. One way of providing such a threshold would be to base it on a moving average over time. On the other hand, the quantity being compared to a threshold, fixed or otherwise, could itself be statistically derived, for example, deriving the ratio of the average number of SYN requests to SYN_ACKs in some statistically significant set of packet data. The use of a "threshold" alone is simply not determinative of whether a detection method is or is not statistical in nature.

17. A type of rudimentary threshold analysis can be used to detect suspicious activity in the context of certain innately suspicious events. For example, a failed login attempt as to a certain type of user account is inherently suspicious. And whereas a single failed login may be ignored as accidental, a sequence, of say, three failed logins is commonly understood by those of skill in the art to be sufficiently suspicious, in and of itself, to require further action. Comparing a count of failed login attempts to a threshold level is considered signature-based detection because it involves *a priori* knowledge of known suspicious activity. In fact, all of the examples given in the patent as lending themselves to "rudimentary, inexpensive signature analysis"—fingers, pings, or failed login requests to accounts such as guest, demo, visitor, anonymous FTP, or employees who have departed the company [Ex. A at 7:50-54][1]—are events that, as individual occurrences, would be considered innately suspicious by a security professional. As such, a very simple threshold count can be used to determine if further action is warranted when dealing with such unique events.

18. Statistical detection methods, as taught by the patents, become necessary when detection is based on "events", in this case individual network packets, that are in and of themselves benign and inherently non-suspicious. For example, individual SYN requests and SYN ACKs are inherently benign. However, an inordinate, unusual spike in the number of unacknowledged SYN requests would be considered suspicious. Detecting SYN Flood attacks requires that the detection engine infer suspicious activity by analyzing aggregate statistics, such as the ratio of SYN Requests to SYN ACKs over time or an accumulated number of packets. Such an analysis is a statistical detection method because it is only by analysis of the statistically significant aggregate of activity that its suspicious nature may be inferred.

19. Alerting on failed login attempts is an example that uses a non-statistical threshold comparison. Discussion of such a method has nothing to do with statistical

[1] All exhibits are attached to this declaration.

analysis, such as the detecting of SYN floods in Symantec's accused products. Failed login attempts are inherently suspicious, and the patent specification lists failed login attempts as one a of a number of specific event types that give rise to suspicion merely by occurring more than a threshold number of times. [*See* Ex. B at 7:52-63]. Also, the specification suggests a different algorithm for detecting failed logins than for detecting SYN Floods because the preferred embodiment uses the signature engine 24 to detect suspicious activity based on failed log-in attempts, but uses profile engine 22, applying statistical methods, to detect SYN Floods.

20. I have read Dr. Hansen's report on non-infringement by the Symantec products. [Ex. C]. Having read Dr. Hansen's report, I am still of the opinion, exactly as stated in my opening report on infringement, that Symantec's accused products perform statistical detection methods as claimed in the claims of the '212 patent and in claim 7 of the '615 patent. I am also still of the opinion that the REDACTED compare long-term and short-term statistical profiles as claimed in the '338 patent.

REDACTED

REDACTED

Response to new assertions in ISS's Motion Regarding Non-Infringement

24. I have reviewed ISS's Motion for Summary Judgment that the Asserted Claims of the SRI Patents-in-suit are not Infringed or, in the Alternative, are Invalid and the Declarations in Support Thereof.

REDACTED

Displaying events at a user interface console is not "integration" as that term is used in the claims of the patents-in-suit.

25.

REDACTED

The Disclosure of the "Ji-Nao" Prior Art

27. I understand that the Defendants assert that Ji-Nao anticipates the hierarchical claims of the '203, '615, and '212 patents. The Defendants base this assertion on Fig. 1 of the *Ji-Nao Report*, SYM_P_0070549, and text from that Report ,

SYM_P_0070577]. But the Ji-Nao project did not attempt to achieve hierarchical correlation. The *Ji-Nao Report* proposes that the disclosed system, which performs detection/analysis functions in a local subsystem, could be "extended to a global level and correlate intrusion events among several routers," [Ex. G at SYM_P_0070548], but acknowledges that such extension "is not within the scope of this project." *Id.* Although hierarchical correlation was a goal of the Ji-Nao researchers, as I have discussed in my Rebuttal Report on Validity, Frank Jou, the principal investigator of the Ji-Nao project, testified that the Ji-Nao system did not attain that goal. [*See* Ex. J at ¶ 104, Ex. H at 172:1-173:11, 170:18-171:2]. Ji-Nao does not anticipate the hierarchical claims of the patents in suit because it does not perform hierarchical correlation.

28. I understand further that the Defendants have characterized my statements that Ji-Nao "reacted to network packets" as conceding that Ji-Nao receives packets, and thus anticipates the claims of the '338 patent. The '338 patent discloses and claims receiving network packets and building profiles from measures of the network packets. As stated in my Rebuttal Report on Validity, the Ji-Nao intrusion analysis function does not directly receive network traffic data (i.e., network packets). [Ex. J at ¶ 101]. Instead, Ji-Nao analyzes logs of data generated by network routers. [*Id.*]. Some of the router audit logs reflect the impact of certain types of data packets on the router that received them. [*Id.* at ¶ 102]. Ji-Nao does not receive or analyze the data packets themselves, and therefore cannot build profiles from measures of the network packets. Instead, Ji-Nao builds profiles from measures of audit log data. This is clearly stated in the Ji-Nao Report's discussion of measures: "[w]e would classify the Ji-Nao measures into two groups: activity intensity and audit record distribution measures. . . . These measures can detect bursts of activity or prolonged activity that is abnormal, primarily based on the volume of audit data generated. The audit record distribution measure determines whether, for recently observed activity (say, the last few hundred audit records received), the types of actions being generated across neighbors are normal." [Ex. G at

SYM_P_0070564]. My statement that Ji-Nao "reacted to network packets" was not a concession that Ji-Nao receives network packets and builds profiles from measures of those packets. Consistent with the Ji-Nao Report and my prior testimony, Ji-Nao "reacts" to network packets only in the sense that it analyzes audit logs that, among other things, represent the impact on routers of certain network packets received by those routers, and builds profiles from measures of those audit logs. Because Ji-Nao does not, however, base its intrusion detection analysis on or build profiles from measures of packets, it does not disclose every limitation of the claims of the '338 patent. Accordingly, Ji-Nao does not anticipate the '338 patent.

I declare under penalty of perjury that the foregoing is true and accurate. Executed this 30th day of June, 2006, in University Park, Pennsylvania.

George Kesidis

CERTIFICATE OF SERVICE

I hereby certify that on July 10, 2006, I electronically filed the **REDACTED – DECLARATION OF DR. GEORGE KESIDIS IN SUPPORT OF SRI INTERNATIONAL, INC.'S RESPONSES TO: (1) DEFENDANTS' JOINT MOTION TO LIMIT THE TESTIMONY OF EXPERT, DR. GEORGE KESIDIS, UNDER FEDERAL RULE OF EVIDENCE 702; (2) DEFENDANTS' JOINT MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. §§ 102 AND 103; (3) SYMANTEC'S MOTION FOR SUMMARY JUDGMENT OF NON-INFRINGEMENT; AND (4) ISS'S MOTION FOR SUMMARY JUDGMENT THAT THE ASSERTED CLAIMS OF THE SRI PATENTS-IN-SUIT ARE NOT INFRINGED OR, IN THE ALTERNATIVE, ARE INVALID** with the Clerk of Court the attached document using CM/ECF which will send electronic notification of such filing(s) to the following Delaware counsel.

Richard L. Horwitz Potter Anderson & Corroon LLP Hercules Plaza 1313 North Market Street, 6th Floor P.O. Box 951 Wilmington, DE 19899	Attorneys for Defendant-Counterclaimant Internet Security Systems, Inc., a Delaware corporation, and Internet Security Systems, Inc., a Georgia corporation
Richard K. Herrmann Morris James Hitchens & Williams PNC Bank Center 222 Delaware Avenue, 10th Floor P.O. Box 2306 Wilmington, DE 19899-2306	Attorneys for Defendant-Counterclaimant Symantec Corporation

/s/ *John F. Horvath*
John F. Horvath

80034097.doc

A

US006321338B1

(12) **United States Patent**
Porras et al.

(10) **Patent No.: US 6,321,338 B1**
(45) **Date of Patent: Nov. 20, 2001**

(54) **NETWORK SURVEILLANCE**

(75) Inventors: **Phillip A. Porras**, Mountain View; **Alfonso Valdes**, San Carlos, both of CA (US)

(73) Assignee: **SRI International**, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/188,739**

(22) Filed: **Nov. 9, 1998**

(51) **Int. Cl.**[7] **G06F 11/30**; G06F 12/14
(52) **U.S. Cl.** **713/201**; 709/224
(58) **Field of Search** 713/201; 709/224

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,672,609	6/1987	Humphrey et al.	371/21
4,773,028	9/1988	Tallman	364/550
5,210,704	5/1993	Husseiny	364/551.01
5,539,659 *	7/1996	McKee et al.	709/224
5,557,742	9/1996	Smaha et al.	395/186
5,706,210 *	1/1998	Kumano et al.	709/224
5,790,799 *	8/1998	Mogul	709/224
5,974,237 *	10/1999	Shurmer et al.	709/224
6,009,467 *	12/1999	Ratcliff et al.	709/224

OTHER PUBLICATIONS

Debar et al., "A Neural Network Component for an Intrusion Detection System,"© 1992 IEEE.
Denning et al., "Prototype IDES: A Real–Time Intrusion–Detection Expert System," SRI Project ECU 7508, SRI International, MenloPark, California, Aug. 1987.
Denning et al., "Requirements and Model For IDES—A Real–Time Intrusion–Detection Expert System," SRI Project 6169, SRI International, Menlo Park, CA, Aug. 1985.
Denning, "An Intrusion–Detection Model," SRI International, Menlo Park, CA, Technical Report CSL–149, Nov. 1985.
Dowell, "The Computerwatch Data Reduction Tool," AT&T Bell Laboratories, Whippany, New Jersey.
Fox et al., "A Neural Network Approach Towards Intrusion Detection," Harris Corporation, Government Information Systems Division, Melbourne, FL, Jul. 2, 1990.
Garvey et al., "Model–Based Intrusion Detection," Proceedings of the 14th National Computer Security Conference, Washington, DC, Oct. 1991.
Ilgun et al., State Transition Analysis: A Rule–Based Intrusion Detection Approach, IEEE Transactions on Software Engineering, vol. 21, No. 3, Mar. 1995.
Javitz et al., "The SRI IDES Statistical Anomaly Detector," Proceedings, 1991 IEEE Symposium on Security and Privacy, Oakland, California, May 1991.
Liepins et al., "Anomaly Detection: Purpose and Framework," US DOE Office of Safeguards and Security.
Lunt et al., "An Expert System to Classify and Sanitize Text," SRI International, Computer Science Laboratory, Menlo Park, CA.

(List continued on next page.)

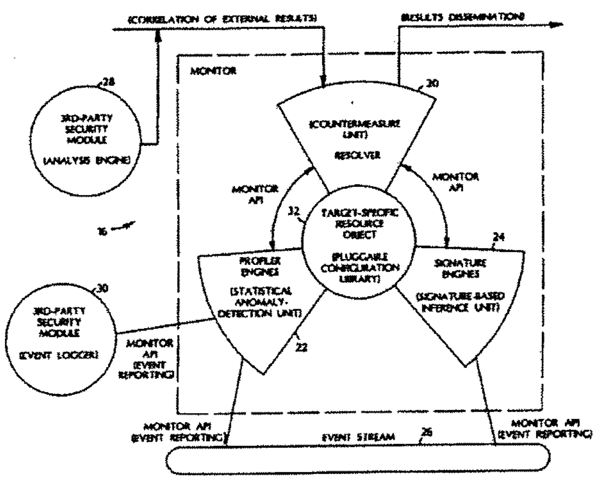
Primary Examiner—Thomas M. Heckler
(74) *Attorney, Agent, or Firm*—Fish & Richardson P.C.

(57) **ABSTRACT**

A method of network surveillance includes receiving network packets handled by a network entity and building at least one long-term and a least one short-term statistical profile from a measure of the network packets that monitors data transfers, errors, or network connections. A comparison of the statistical profiles is used to determine whether the difference between the statistical profiles indicates suspicious network activity.

27 Claims, 5 Drawing Sheets

Microfiche Appendix Included
(10 Microfiche, 956 Pages)



OTHER PUBLICATIONS

Lunt, "A Survey of Intrusion Detection Techniques," Computers & Security, 12 (1993) 405–418.

Lunt, "Automated Audit Trail Analysis and Intrusion Detection: A Survey," Proceedings of the 11th National Computer Security Conference, Baltimore, MD, Oct. 1988.

Lunt et al, "Knowledge–Based Intrusion Detection".

Lunt et al., "A Prototype Real–Time Intrusion–Detection Expert System," Proceedings of the 1988 IEEE Symposium on Security and Privacy, Apr. 1988.

Porras et al., EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances, 20th NISSC—Oct. 9, 1997.

Porras et al., Penetration State Transition Analysis A Rule–Based Intrusion Detection Approach, © 1992 IEEE.

Sebring et al., Expert Systems in Intrusion Detection: A Case Study.

Shieh et al., A Pattern–Oriented Intrusion–Detection Model and Its Applications © 1991 IEEE.

Smaha, "Haystack: An Intrusion Detection System," © 1988 IEEE Computer Society Press: Proceedings of the Fourth Aerospace Computer Security Applications Conference, 1988, pp. 37–44.

Snapp, "Signature Analysis and Communication Issues in a Distributed Intrusion Detection System," Thesis 1991.

Snapp et al., "DIDS (Distributed Intrusion Detection System)—Motivation, Architecture, and An Early Protoype, "Computer Security Laboratory, Division of Computer Science, Univ. of California, Davis, Davis, CA.

Tener, "AI & 4GL: Automated Detection and Investigation Tools," Computer Security in the Age of Information, Proceedings of the Fifth IFIP International Conference on Computer Security, W.J. Caelli (ed.).

Teng et al., "Adaptive Real–Time Anomaly Detection Using Inductively Generated Sequential Patterns," © 1990.

Vacccaro et al., "Detection of Anomalous Computer Session Activity," © 1989 IEEE.

Weiss, "Analysis of Audit and Protocol Data using Methods from Artificial Intelligence," Siemens AG, Munich, West Germany.

Winkler, "A UNIX Prototype for Intrusion and Anomaly Detection in Secure Networks," © Planning Research Corp. 1990.

Jarvis et al., The NIDES Statistical Component Description and Justification, SRI International Annual Report A010, Mar. 7, 1994.

Debar, et al., "Towards a Taxonomy of Intrusion–Detection Systems," Computers Networks 31 (1999), 805–822.

Garvey, et al., "An Inference Technique for Integrating Knowledge from Disparate Sources," Proc. IJCAI, Vancouver, B.C., Aug., 1981, 319–325.

Kaven, "The Digital Doorman," PC Magazine, Nov. 16, 1999.

Lindqvist, et al., "Detecting Computer and Network Misuse Through the Production–Based Expert System Toolset (P–BEST)," Oct. 25, 1998.

* cited by examiner

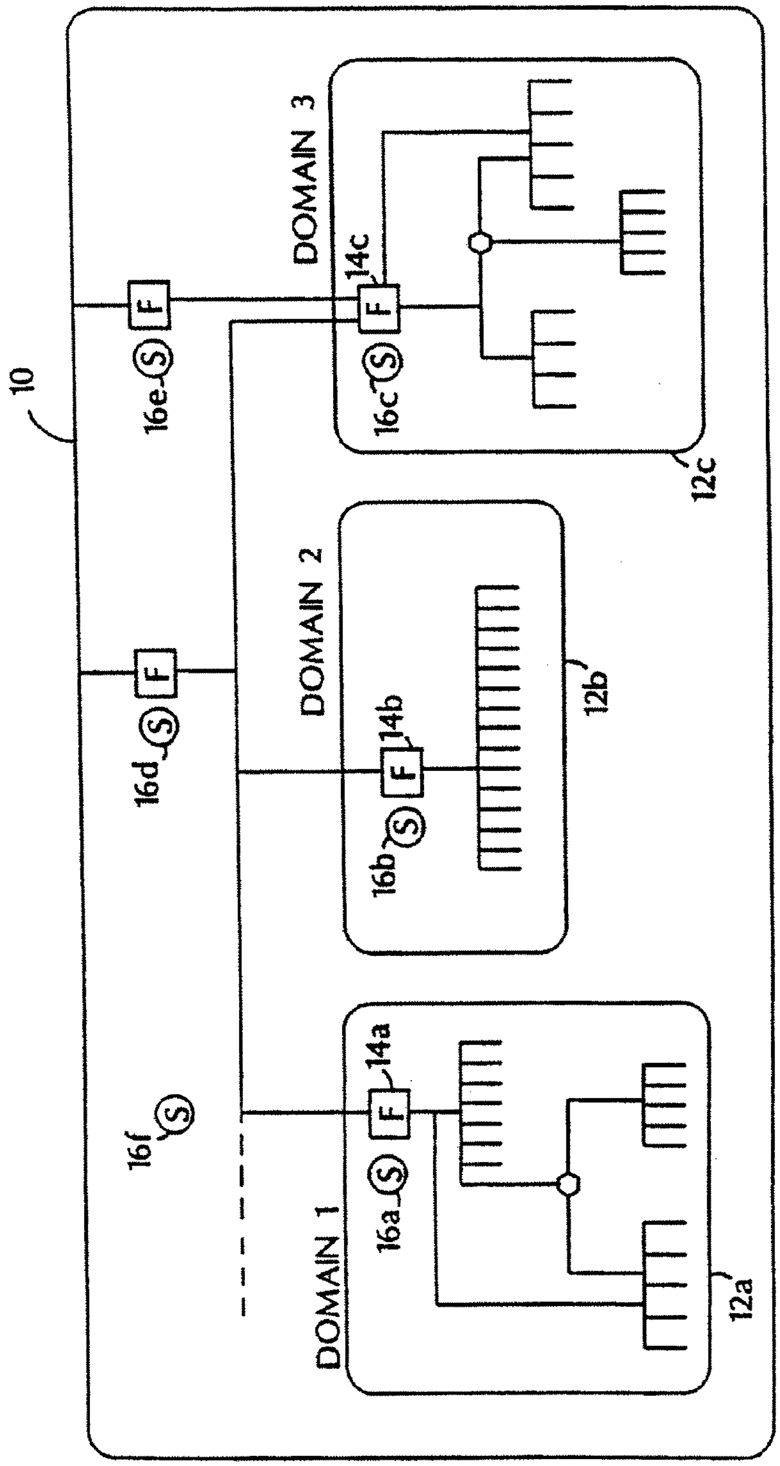

10
DOMAIN 1
DOMAIN 2
DOMAIN 3
12a
12b
12c
14a
14b
14c
16a
16b
16c
16d
16e
16f
F
S

FIG. 1

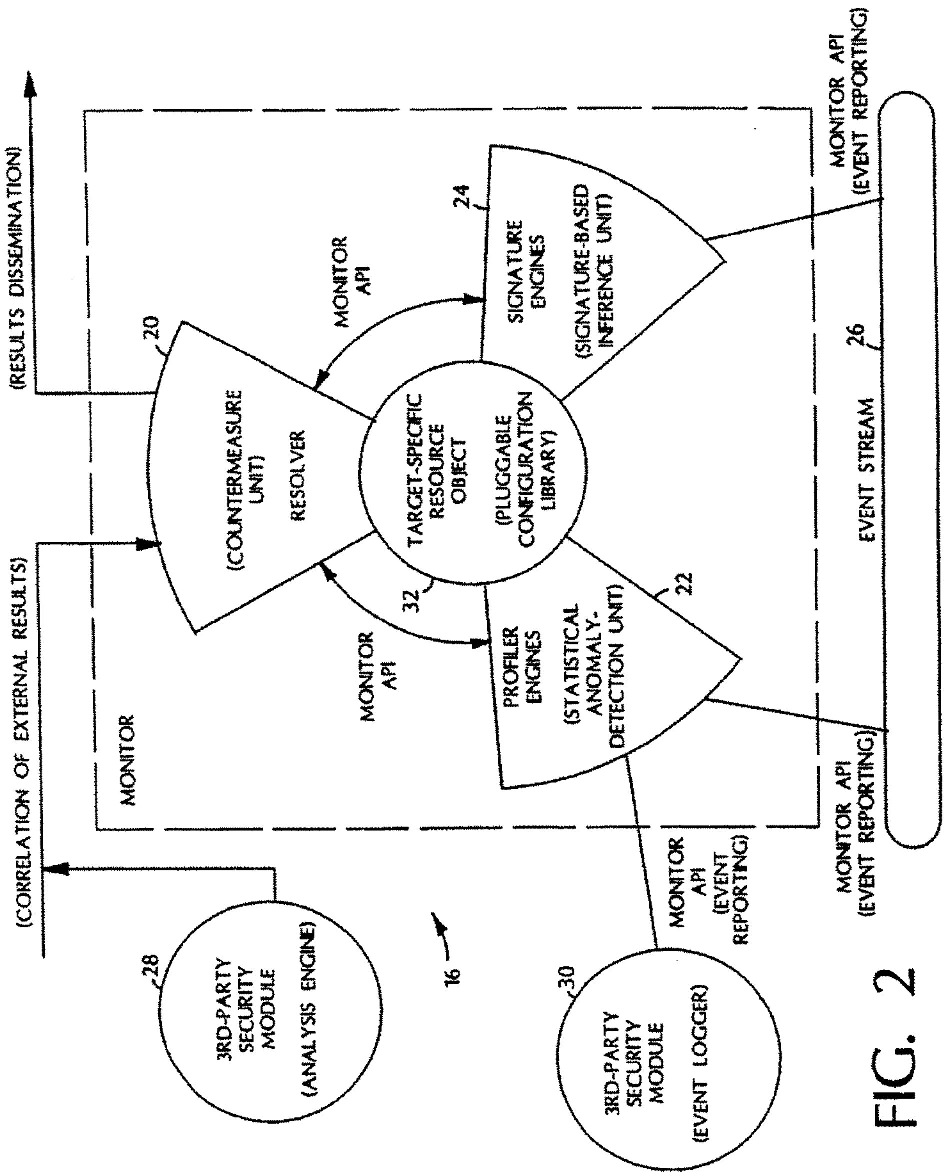

(RESULTS DISSEMINATION)
(CORRELATION OF EXTERNAL RESULTS)
MONITOR
20
(COUNTERMEASURE UNIT)
RESOLVER
MONITOR API
24
SIGNATURE ENGINES
(SIGNATURE-BASED INFERENCE UNIT)
TARGET-SPECIFIC RESOURCE OBJECT
(PLUGGABLE CONFIGURATION LIBRARY)
32
MONITOR API
PROFILER ENGINES
(STATISTICAL ANOMALY-DETECTION UNIT)
22
MONITOR API (EVENT REPORTING)
26
EVENT STREAM
MONITOR API (EVENT REPORTING)
MONITOR API (EVENT REPORTING)
28
3RD-PARTY SECURITY MODULE
(ANALYSIS ENGINE)
16
30
3RD-PARTY SECURITY MODULE
(EVENT LOGGER)

FIG. 2

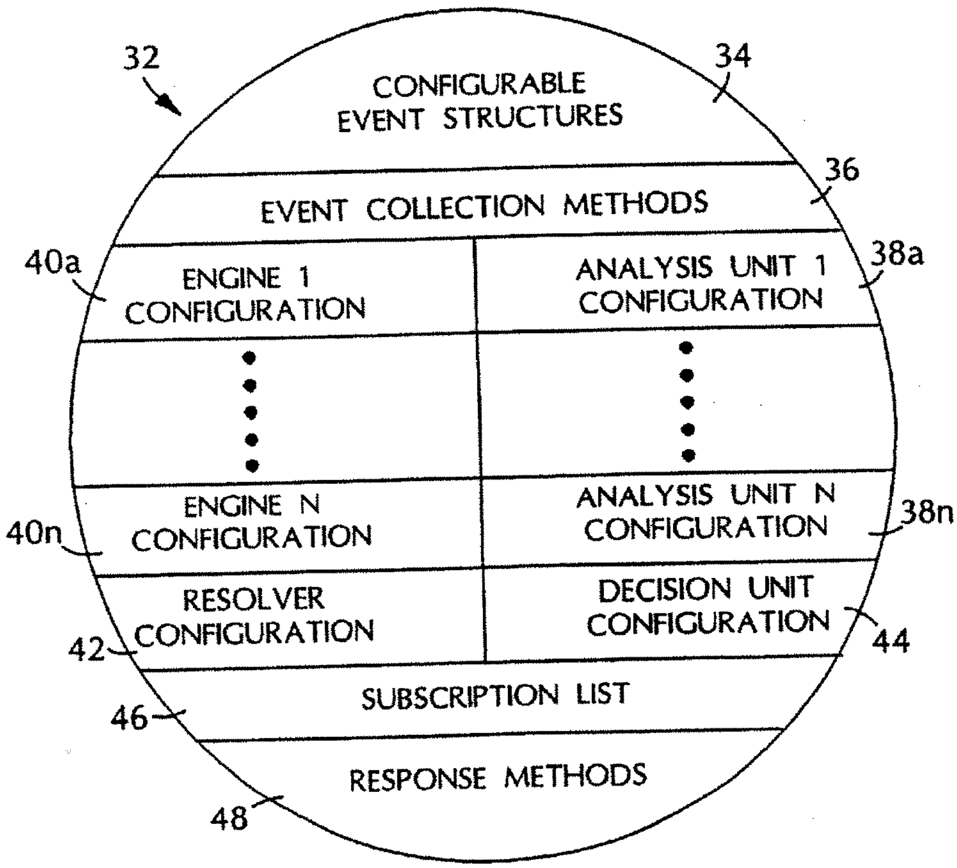
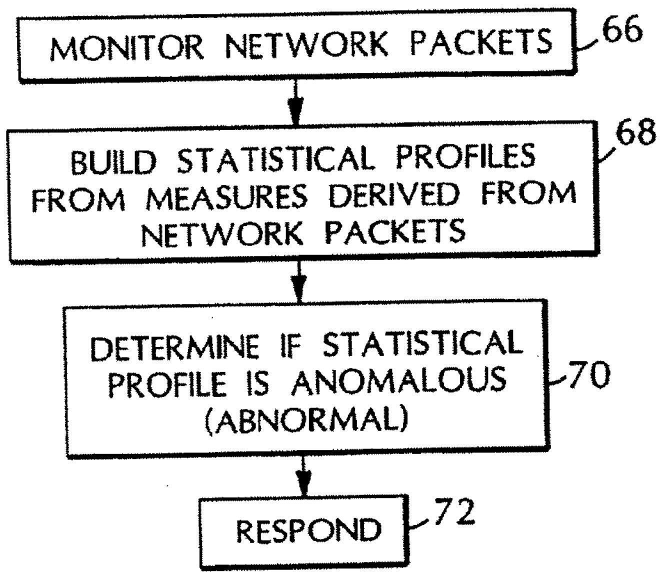
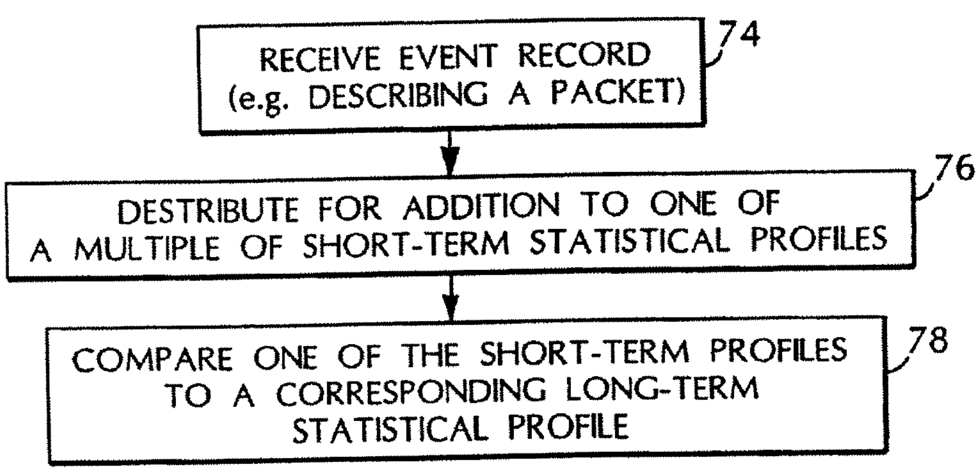

32
34
CONFIGURABLE EVENT STRUCTURES
36
EVENT COLLECTION METHODS
40a
ENGINE 1 CONFIGURATION
ANALYSIS UNIT 1 CONFIGURATION
38a
40n
ENGINE N CONFIGURATION
ANALYSIS UNIT N CONFIGURATION
38n
RESOLVER CONFIGURATION
42
DECISION UNIT CONFIGURATION
44
46
SUBSCRIPTION LIST
RESPONSE METHODS
48

FIG. 3



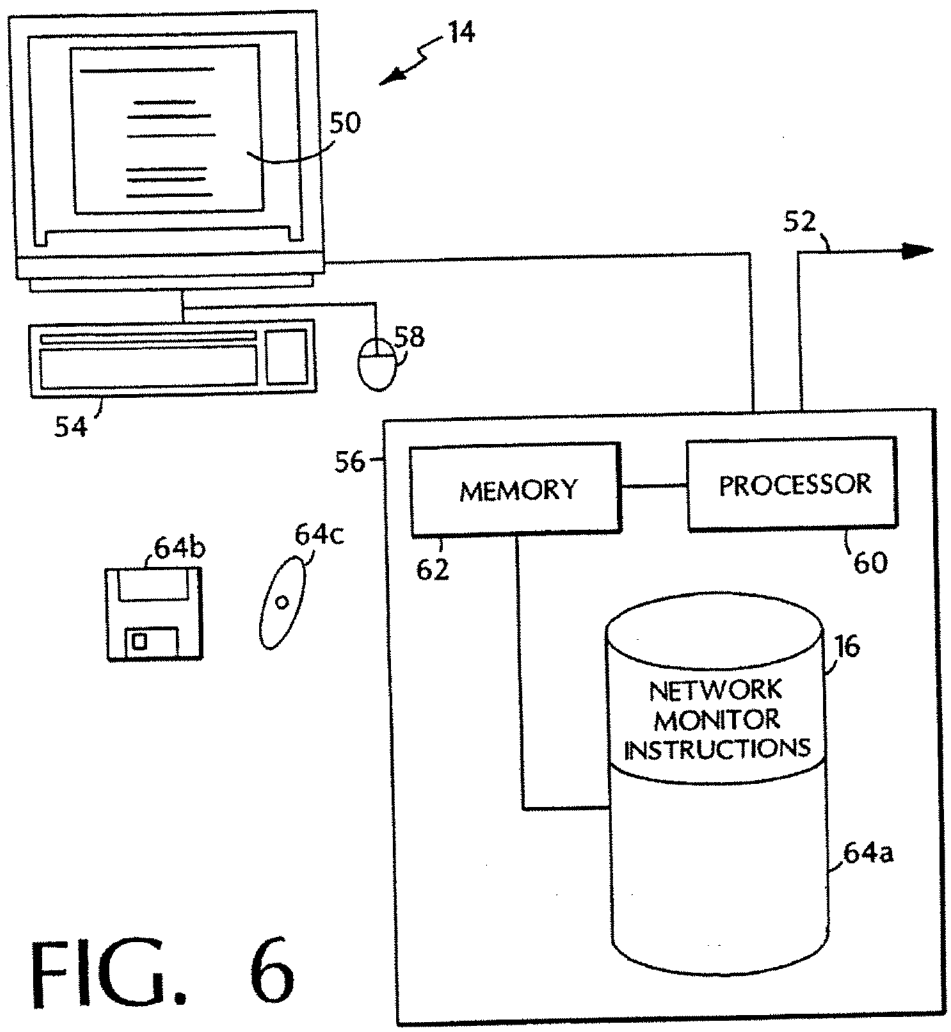
MONITOR NETWORK PACKETS
66
BUILD STATISTICAL PROFILES FROM MEASURES DERIVED FROM NETWORK PACKETS
68
DETERMINE IF STATISTICAL PROFILE IS ANOMALOUS (ABNORMAL)
70
RESPOND
72

FIG. 4



RECEIVE EVENT RECORD (e.g. DESCRIBING A PACKET)
74
DESTRIBUTE FOR ADDITION TO ONE OF A MULTIPLE OF SHORT-TERM STATISTICAL PROFILES
76
COMPARE ONE OF THE SHORT-TERM PROFILES TO A CORRESPONDING LONG-TERM STATISTICAL PROFILE
78

FIG. 5



14
50
52
58
54
56
MEMORY
PROCESSOR
64b
64c
62
60
16
NETWORK
MONITOR
INSTRUCTIONS
64a

FIG. 6

NETWORK SURVEILLANCE

REFERENCE TO GOVERNMENT FUNDING

This invention was made with Government support under Contract Number F30602-96-C-0294 awarded by DARPA. The Government has certain rights in this invention.

REFERENCE TO APPENDIX

BACKGROUND

The invention relates to computer networks.

Computer networks offer users ease and efficiency in exchanging information. Networks tend to include conglomerates of integrated commercial and custom-made components, interoperating and sharing information at increasing levels of demand and capacity. Such varying networks manage a growing list of needs including transportation, commerce, energy management, communications, and defense.

Unfortunately, the very interoperability and sophisticated integration of technology that make networks such valuable assets also make them vulnerable to attack, and make dependence on networks a potential liability. Numerous examples of planned network attacks, such as the Internet worm, have shown how interconnectivity can be used to spread harmful program code. Accidental outages such as the 1980 ARPAnet collapse and the 1990 AT&T collapse illustrate how seemingly localized triggering events can have globally disastrous effects on widely distributed systems. In addition, organized groups have performed malicious and coordinated attacks against various online targets.

SUMMARY

In general, in one aspect, a method of network surveillance includes receiving network packets (e.g., TCP/IP packets) handled by a network entity and building at least one long-term and at least one short-term statistical profile from at least one measure of the network packets that monitors data transfers, errors, or network connections. A comparison of at least one long-term and at least one short-term statistical profile is used to determine whether the difference between the short-term statistical profile and the long-term statistical profile indicates suspicious network activity.

Embodiments may include one or more of the following features. The measure may monitor data transfers by monitoring network packet data transfer commands, data transfer errors, and/or monitoring network packet data transfer volume. The measure may monitor network connections by monitoring network connection requests, network connection denials, and/or a correlation of network connections requests and network connection denials. The measure may monitor errors by monitoring error codes included in a network packet such as privilege error codes and/or error codes indicating a reason a packet was rejected.

The method may also include responding based on the determining whether the difference between a short-term

statistical profile and a long-term statistical profile indicates suspicious network activity. A response may include altering analysis of network packets and/or severing a communication channel. A response may include transmitting an event record to a network monitor, such as hierarchically higher network monitor and/or a network monitor that receives event records from multiple network monitors.

The network entity may be a gateway, a router, or a proxy server. The network entity may instead be a virtual private network entity (e.g., node).

In general, in another aspect, a method of network surveillance includes monitoring network packets handled by a network entity and building a long-term and multiple short-term statistical profiles of the network packets. A comparison of one of the multiple short-term statistical profiles with the long-term statistical profile is used to determine whether the difference between the short-term statistical profiles and the long-term statistical profile indicates suspicious network activity.

Embodiments may include one or more of the following. The multiple short-term statistical profiles may monitor different anonymous FTP sessions. Building multiple short-term statistical profiles may include deinterleaving packets to identify a short-term statistical profile.

In general, in another aspect, a computer program product, disposed on a computer readable medium, includes instructions for causing a processor to receive network packets handled by a network entity and to build at least one long-term and at least one short-term statistical profile from at least one measure of the network packets that monitors data transfers, errors, or network connections. The instructions compare a short-term and a long-term statistical profile to determine whether the difference between the short-term statistical profile and the long-term statistical profile indicates suspicious network activity.

In general, in another aspect, a method of network surveillance includes receiving packets at a virtual private network entity and statistically analyzing the received packets to determine whether the packets indicate suspicious network activity. The packets may or may not be decrypted before statistical analysis.

Advantages may include one or more of the following. Using long-term and a short-term statistical profiles from measures that monitor data transfers, errors, or network connections protects network components from intrusion. As long-term profiles represent "normal" activity, abnormal activity may be detected without requiring an administrator to catalog each possible attack upon a network. Additionally, the ability to deinterleave packets to create multiple short-term profiles for comparison against a long-term profile enables the system to detect abnormal behavior that may be statistically ameliorated if only a single short-term profile was created.

The scheme of communication network monitors also protects networks from more global attacks. For example, an attack made upon one network entity may cause other entities to be alerted. Further, a monitor that collects event reports from different monitors may correlate activity to identify attacks causing disturbances in more than one network entity.

Additionally, statistical analysis of packets handled by a virtual private network enable detection of suspicious network activity despite virtual private network security techniques such as encryption of the network packets.

Other features and advantages will become apparent from the following description, including the drawings, and from the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a diagram of network monitors deployed in an enterprise.

FIG. 2 is a diagram of a network monitor that monitors an event stream.

FIG. 3 is a diagram of a resource object that configures the network monitor of FIG. 2.

FIG. 4 is a flowchart illustrating network surveillance.

FIG. 5 is a flowchart illustrating multiple short-term statistical profiles for comparison against a single long-term statistical profile.

FIG. 6 is a diagram of a computer platform suitable for deployment of a network monitor.

DETAILED DESCRIPTION

Referring to FIG. 1, an enterprise **10** includes different domains **12a–12c**. Each domain **12a–12c** includes one or more computers offering local and network services that provide an interface for requests internal and external to the domain **12a–12c**. Network services include features common to many network operating systems such as mail, HTTP, FTP, remote login, network file systems, finger, Kerberos, and SNMP. Some domains **12a–12c** may share trust relationships with other domains (either peer-to-peer or hierarchical). Alternatively, domains **12a–12c** may operate in complete mistrust of all others, providing outgoing connections only or severely restricting incoming connections. Users may be local to a single domain or may possess accounts on multiple domains that allow them to freely establish connections throughout the enterprise **10**.

As shown, the enterprise **10** includes dynamically deployed network monitors **16a–16f** that analyze and respond to network activity and can interoperate to form an analysis hierarchy. The analysis hierarchy provides a framework for the recognition of more global threats to interdomain connectivity, including coordinated attempts to infiltrate or destroy connectivity across an entire network enterprise **10**. The hierarchy includes service monitors **16a–16c**, domain monitors **16d–16e**, and enterprise monitors **16f**.

Service monitors **16a–16c** provide local real-time analysis of network packets (e.g., TCP/IP packets) handled by a network entity **14a–14c**. Network entities include gateways, routers, firewalls, or proxy servers. A network entity may also be part of a virtual private network. A virtual private network (VPN) is constructed by using public wires to connect nodes. For example, a network could use the Internet as the medium for transporting data and use encryption and other security mechanisms to ensure that only authorized users access the network and that the data cannot be intercepted. A monitor **16a–16f** can analyze packets both before and after decryption by a node of the virtual private network.

Information gathered by a service monitor **16a–16c** can be disseminated to other monitors **16a–16f**, for example, via a subscription-based communication scheme. In a subscription-based scheme client monitors subscribe to receive analysis reports produced by server monitors. As a monitor **16a–16f** produces analysis reports, the monitor **16a–16f** disseminates these reports asynchronously to subscribers. Through subscription, monitors **16a–16f** distributed throughout a large network are able to efficiently disseminate reports of malicious activity without requiring the overhead of synchronous polling.

Domain monitors **16d–16e** perform surveillance over all or part of a domain **12a–12c**. Domain monitors **16d–16e** correlate intrusion reports disseminated by individual service monitors **16a–16c**, providing a domain-wide perspective of activity (or patterns of activity). In addition to domain surveillance, domain monitors **16a–16c** can reconfigure system parameters, interface with other monitors beyond a domain, and report threats against a domain **12a–12c** to administrators. Domain monitors **16d–16e** can subscribe to service monitors **16a–16c**. Where mutual trust among domains **12a–12c** exists, domain monitors **16d–16e** may establish peer relationships with one another. Peer-to-peer subscription allows domain monitors **16d–16e** to share analysis reports produced in other domains **12a–12c**. Domain monitors **16d–16e** may use such reports to dynamically sensitize their local service monitors **16a–16c** to malicious activity found to be occurring outside a domain **12a–12c**. Domain monitors **16d–16e** may also operate within an enterprise hierarchy where they disseminate analysis reports to enterprise monitors **16f** for global correlation.

Enterprise monitors **16f** correlate activity reports produced across the set of monitored domains **12a–12c**. Enterprise **10** surveillance may be used where domains **12a–12c** are interconnected under the control of a single organization, such as a large privately owned WAN (Wide Area Network). The enterprise **10**, however, need not be stable in its configuration or centrally administered. For example, the enterprise **10** may exist as an emergent entity through new interconnections of domains **12a–12c**. Enterprise **10** surveillance is very similar to domain **12a–12c** surveillance: an enterprise monitor **16f** subscribes to various domain monitors **16d–16e**, just as the domain monitors **16d–16e** subscribed to various service monitors **16a–16c**. The enterprise monitor **16f** (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain. As an enterprise monitor **16f** recognizes commonalities in intrusion reports across domains (e.g., the spreading of a worm or a mail system attack repeated throughout the enterprise **10**), the monitor **16f** can help domains **12a–12c** counter the attack and can sensitize other domains **12a–12c** to such attacks before they are affected. Through correlation and sharing of analysis reports, reports of problems found by one monitor **16a–16f** may propagate to other monitors **16a–16f** throughout the network. Interdomain event analysis is vital to addressing more global, information attacks against the entire enterprise **10**.

Referring to FIG. 2, each monitor **16** includes one or more analysis engines **22**, **24**. These engines **22**, **24** can be dynamically added, deleted, and modified as necessary. In the dual-analysis configuration shown, a monitor **16** instantiation includes a signature analysis engine **22** and a statistical profiling engine **24**. In general, a monitor **16** may include additional analysis engines that may implement other forms of analysis. A monitor **16** also includes a resolver **20** that implements a response policy and a resource object **32** that configures the monitor **16**. The monitors **16** incorporate an application programmers' interface (API) that enhances encapsulation of monitor functions and eases integration of third-party intrusion-detection tools **28**, **30**.

Each monitor **16** can analyze event records that form an event stream. The event stream may be derived from a variety of sources such as TCP/IP network packet contents or event records containing analysis reports disseminated by other monitors. For example, an event record can be formed from data included in the header and data segment of a network packet. The volume of packets transmitted and

received, however, dictates careful assessment of ways to select and organize network packet information into event record streams.

Selection of packets can be based on different criteria. Streams of event records can be derived from discarded traffic (i.e., packets not allowed through the gateway because they violate filtering rules), pass-through traffic (i.e., packets allowed into the internal network from external sources), packets having a common protocol (e.g., all ICMP (Internet Control Message Protocol) packets that reach the gateway), packets involving network connection management (e.g., SYN, RESET, ACK, [window resize]), and packets targeting ports to which an administrator has not assigned any network service and that also remain unblocked by the firewall. Event streams may also be based on packet source addresses (e.g., packets whose source addresses match well-known external sites such as satellite offices or have raised suspicion from other monitoring efforts) or destination addresses (e.g., packets whose destination addresses match a given internal host or workstation). Selection can also implement application-layer monitoring (e.g., packets targeting a particular network service or application). Event records can also be produced from other sources of network packet information such as report logs produced by network entities. Event streams can be of very fine granularity. For example, a different stream might be derived for commands received from different commercial web-browsers since each web-browser produces different characteristic network activity.

A monitor **16** can also construct interval summary event records, which contain accumulated network traffic statistics (e.g., number of packets and number of kilobytes transferred). These event records are constructed at the end of each interval (e.g., once per N seconds). Event records are forwarded to the analysis engines **22**, **24** for analysis.

The profile engine **22** can use a wide range of multivariate statistical measures to profile network activity indicated by an event stream. A statistical score represents how closely currently observed usage corresponds to the established patterns of usage. The profiler engine **22** separates profile management and the mathematical algorithms used to assess the anomaly of events. The profile engine **22** may use a statistical analysis technique described in A. Valdes and D. Anderson, "Statistical Methods for Computer Usage Anomaly Detection Using NIDES", Proceedings of the Third International Workshop on Rough Sets and Soft Computing, January 1995, which is incorporated by reference in its entirety. Such an engine **22** can profile network activity via one or more variables called measures. Measures can be categorized into four classes: categorical, continuous, intensity, and event distribution measures.

Categorical measures assume values from a discrete, nonordered set of possibilities. Examples of categorical measures include network source and destination addresses, commands (e.g., commands that control data transfer and manage network connections), protocols, error codes (e.g., privilege violations, malformed service requests, and malformed packet codes), and port identifiers. The profiler engine **22** can build empirical distributions of the category values encountered, even if the list of possible values is open-ended. The engine **22** can have mechanisms for "aging out" categories whose long-term probabilities drop below a threshold.

Continuous measures assume values from a continuous or ordinal set. Examples include inter-event time (e.g., difference in time stamps between consecutive events from the same stream), counting measures such as the number of errors of a particular type observed in the recent past, the volume of data transfers over a period of time, and network traffic measures (number of packets and number of kilobytes). The profiler engine **22** treats continuous measures by first allocating bins appropriate to the range of values of the underlying measure, and then tracking the frequency of observation of each value range. In this way, multi-modal distributions are accommodated and much of the computational machinery used for categorical measures is shared. Continuous measures are useful not only for intrusion detection, but also to support the monitoring of the health and status of the network from the perspective of connectivity and throughput. For example, a measure of traffic volume maintained can detect an abnormal loss in the data rate of received packets when this volume falls outside historical norms. This sudden drop can be specific both to the network entity being monitored and to the time of day (e.g., the average sustained traffic rate for a major network artery is much different at 11:00 a.m. than at midnight).

Intensity measures reflect the intensity of the event stream (e.g., number of ICMP packets) over specified time intervals (e.g., 1 minute, 10 minutes, and 1 hour). Intensity measures are particularly suited for detecting flooding attacks, while also providing insight into other anomalies.

Event distribution measures are meta-measures that describes how other measures in the profile are affected by each event. For example, an "ls" command in an FTP session affects the directory measure, but does not affect measures related to file transfer. This measure is not interesting for all event streams. For example, all network-traffic event records affect the same measures (number of packets and kilobytes) defined for that event stream, so the event distribution does not change. On the other hand, event distribution measures are useful in correlative analysis performed by a monitor **16a–16f** that receives reports from other monitors **16a–16f**.

The system maintains and updates a description of behavior with respect to these measure types in an updated profile. The profile is subdivided into short-term and long-term profiles. The short-term profile accumulates values between updates, and exponentially ages (e.g., weighs data based on how long ago the data was collected) values for comparison to the long-term profile. As a consequence of the aging mechanism, the short-term profile characterizes recent activity, where "recent" is determined by a dynamically configurable aging parameters. At update time (typically, a time of low system activity), the update function folds the short-term values observed since the last update into the long-term profile, and the short-term profile is cleared. The long-term profile is itself slowly aged to adapt to changes in subject activity. Anomaly scoring compares related attributes in the short-term profile against the long-term profile. As all evaluations are done against empirical distributions, no assumptions of parametric distributions are made, and multi-modal and categorical distributions are accommodated. Furthermore, the algorithms require no a priori knowledge of intrusive or exceptional activity.

The statistical algorithm adjusts a short-term profile for the measure values observed in the event record. The distribution of recently observed values is compared against the long-term profile, and a distance between the two is obtained. The difference is compared to a historically adaptive deviation. The empirical distribution of this deviation is transformed to obtain a score for the event. Anomalous events are those whose scores exceed a historically adaptive score threshold based on the empirical score distribution.

This nonparametric approach handles all measure types and makes no assumptions on the modality of the distribution for continuous measures.

Profiles are provided to the computational engine as classes defined in the resource object 32. The mathematical functions for anomaly scoring, profile maintenance, and updating do not require knowledge of the data being analyzed beyond what is encoded in the profile class. Event collection interoperability supports translation of the event stream to the profile and measure classes. At that point, analysis for different types of monitored entities is mathematically similar. This approach imparts great flexibility to the analysis in that fading memory constants, update frequency, measure type, and so on are tailored to the network entity being monitored.

The measure types described above can be used individually or in combination to detect network packet attributes characteristic of intrusion. Such characteristics include large data transfers (e.g., moving or downloading files), an increase in errors (e.g., an increase in privilege violations or network packet rejections), network connection activity, and abnormal changes in network volume.

As shown, the monitor 16 also includes a signature engine 24. The signature engine 24 maps an event stream against abstract representations of event sequences that are known to indicate undesirable activity. Signature-analysis objectives depend on which layer in the hierarchical analysis scheme the signature engine operates. Service monitor 16*a*–16*c* signature engines 24 attempt to monitor for attempts to penetrate or interfere with the domain's operation. The signature engine scans the event stream for events that represent attempted exploitations of known attacks against the service, or other activity that stands alone as warranting a response from the monitor. Above the service layer, signature engines 24 scan the aggregate of intrusion reports from service monitors in an attempt to detect more global coordinated attack scenarios or scenarios that exploit interdependencies among network services. Layering signature engine analysis enables the engines 24 to avoid misguided searches along incorrect signature paths in addition to distributing the signature analysis.

A signature engines 24 can detect, for example, address spoofing, tunneling, source routing, SATAN attacks, and abuse of ICMP messages ("Redirect" and "Destination Unreachable" messages in particular). Threshold analysis is a rudimentary, inexpensive signature analysis technique that records the occurrence of specific events and, as the name implies, detects when the number of occurrences of that event surpasses a reasonable count. For example, monitors can encode thresholds to monitor activity such as the number of fingers, pings, or failed login requests to accounts such as guest, demo, visitor, anonymous FTP, or employees who have departed the company.

Signature engine 24 can also examine the data portion of packets in search of a variety of transactions that indicate suspicious, if not malicious, intentions by an external client. The signature engine 24, for example, can parse FTP traffic traveling through the firewall or router for unwanted transfers of configuration or specific system data, or anonymous requests to access non-public portions of the directory structure. Similarly, a monitor can analyze anonymous FTP sessions to ensure that the file retrievals and uploads/modifications are limited to specific directories. Additionally, signature analysis capability can extend to session analyses of complex and dangerous, but highly useful, services like HTTP or Gopher.

Signature analysis can also scan traffic directed at unused ports (i.e., ports to which the administrator has not assigned a network service). Here, packet parsing can be used to study network traffic after some threshold volume of traffic, directed at an unused port, has been exceeded. A signature engine 24 can also employ a knowledge base of known telltale packets that are indicative of well-known network-service protocol traffic (e.g., FTP, Telnet, SMTP, HTTP). The signature engine 24 then determines whether the unknown port traffic matches any known packet sets. Such comparisons could lead to the discovery of network services that have been installed without an administrator's knowledge.

The analysis engines 22, 24 receive large volumes of events and produce smaller volumes of intrusion or suspicion reports that are then fed to the resolver 20. The resolver 20 is an expert system that receives the intrusion and suspicion reports produced by the analysis engines 22, 24 and reports produced externally by other analysis engines to which it subscribes. Based on these reports, the resolver 20 invokes responses. Because the volume of intrusion and suspicion reports is lower than the volume of events received by the analysis engines 22, 24, the resolver 20 can afford the more sophisticated demands of configuration maintenance and managing the response handling and external interfaces necessary for monitor operation. Furthermore, the resolver 20 adds to extensibility by providing the subscription interface through which third-party analysis tools 28, 30 can interact and participate in the hierarchical analysis scheme.

Upon its initialization, the resolver 20 initiates authentication and subscription sessions with those monitors 16*a*–16*f* whose identities appear in the monitor's 16 subscription-list (46 FIG. 3). The resolver 20 also handles all incoming requests by subscribers, which must authenticate themselves to the resolver 20. Once a subscription session is established with a subscriber monitor, the resolver 20 acts as the primary interface through which configuration requests are received and intrusion reports are disseminated.

Thus, resolvers 20 can request and receive reports from other resolvers at lower layers in the analysis hierarchy. The resolver 20 forwards analysis reports received from subscribees to the analysis engines 22, 24. This tiered collection and correlation of analysis results allows monitors 16*a*–16*f* to represent and profile global malicious or anomalous activity that is not visible locally.

In addition to external-interface responsibilities, the resolver 20 operates as a fully functional decision engine, capable of invoking real-time response measures in response to malicious or anomalous activity reports produced by the analysis engines. The resolver 20 also operates as the center of intramonitor communication. As the analysis engines 22, 24 build intrusion and suspicion reports, they propagate these reports to the resolver 20 for further correlation, response, and dissemination to other monitors 16*a*–16*f*. The resolver 20 can also submit runtime configuration requests to the analysis engines 22, 24, for example, to increase or decrease the scope of analyses (e.g., enable or disable additional signature rules) based on various operating metrics. These configuration requests could be made as a result of encountering other intrusion reports from other subscribers. For example, a report produced by a service monitor 16*a*–16*c* in one domain could be propagated to an enterprise monitor 16*f*, which in turn sensitizes service monitors in other domains to the same activity.

The resolver 20 also operates as the interface mechanism between administrators and the monitor 16. From the per-

spective of a resolver **20**, the administrator interface is simply a subscribing service to which the resolver **20** may submit reports and receive configuration requests. An administrative interface tool can dynamically subscribe and unsubscribe to any of the deployed resolvers **20**, as well as submit configuration requests and asynchronous probes as desired.

The monitors **16***a*–**16***f* incorporate a bidirectional messaging system that uses a standard interface specification for communication within and between monitor elements and external modules. Using this interface specification, third-party modules **28, 30** can communicate with monitors. For example, third-party modules **28** can submit event records to the analysis engines **22, 24** for processing. Additionally, third-party modules **30** may also submit and receive analysis results via the resolver's **20** external interfaces. Thus, third-party modules **28, 30** can incorporate the results from monitors into other surveillance efforts or contribute their results to other monitors **16***a*–**16***f*. Lastly, the monitor's **16** internal API allows third-party analysis engines to be linked directly into the monitor boundary.

The message system operates under an asynchronous communication model for handling results dissemination and processing that is generically referred to as subscription-based message passing. Component interoperation is client/server-based, where a client module may subscribe to receive event data or analysis results from servers. Once a subscription request is accepted by the server, the server module forwards events or analysis results to the client automatically as data becomes available, and may dynamically reconfigure itself as requested by the client's control requests. This asynchronous model reduces the need for client probes and acknowledgments.

The interface supports an implementation-neutral communication framework that separates the programmer's interface specification and the issues of message transport. The interface specification embodies no assumptions about implementation languages, host platform, or a network. The transport layer is architecturally isolated from the internals of the monitors so that transport modules may be readily introduced and replaced as protocols and security requirements are negotiated between module developers. The interface specification involves the definition of the messages that the various intrusion-detection modules must convey to one another and how these messages should be processed. The message structure and content are specified in a completely implementation-neutral context.

Both intramonitor and intermonitor communication employ identical subscription-based client-server models. With respect to intermonitor communication, the resolver **20** operates as a client to the analysis engines, and the analysis engines **22, 24** operate as clients to the event filters. Through the internal message system, the resolver **20** submits configuration requests to the analysis engines **22, 24**, and receives from the analysis engines **22, 24** their analysis results. The analysis engines **22, 24** operate as servers providing the resolver **20** with intrusion or suspicion reports either asynchronously or upon request. Similarly, the analysis engines **22, 24** are responsible for establishing and maintaining a communication link with an event collection method (or event filter) and prompting the reconfiguration of the collection method's filtering semantics when necessary.

Intermonitor communication also operates using the subscription-based hierarchy. A domain monitor **16***d*–**16***e* subscribes to the analysis results produced by service monitors **16***a*–**16***c*, and then propagates its own analytical reports to its parent enterprise monitor **16***f*. The enterprise monitor **16***f* operates as a client to one or more domain monitors **16***d*–**16***e*, allowing them to correlate and model enterprise-wide activity from the domain-layer results. Domain monitors **16***d*–**16***e* operate as servers to the enterprise monitors **16***f*, and as clients to the service monitors **16***a*–**16***c* deployed throughout their domain **12***a*–**12***c*. This message scheme can operate substantially the same if correlation were to continue at higher layers of abstraction beyond enterprise **10** analysis.

Intramonitor and intermonitor programming interfaces are substantially the same. These interfaces can be subdivided into five categories of interoperation: channel initialization and termination, channel synchronization, dynamic configuration, server probing, and report/event dissemination. Clients are responsible for initiating and terminating channel sessions with servers. Clients are also responsible for managing channel synchronization in the event of errors in message sequencing or periods of failed or slow response (i.e., "I'm alive" confirmations). Clients may also submit dynamic configuration requests to servers. For example, an analysis engine **22, 24** may request an event collection method to modify its filtering semantics. Clients may also probe servers for report summaries or additional event information. Lastly, servers may send clients intrusion/suspicion reports in response to client probes or in an asynchronous dissemination mode.

The second part of the message system framework involves specification of a transport mechanism used to establish a given communication channel between monitors **16***a*–**16***f* or possibly between a monitor **16***a*–**16***f* and a third-party security module. All implementation dependencies within the message system framework are addressed by pluggable transport modules. Transport modules are specific to the participating intrusion-detection modules, their respective hosts, and potentially to the network—should the modules require cross-platform interoperation. Instantiating a monitor **16***a*–**16***f* may involve incorporation of the necessary transport module(s) (for both internal and external communication).

The transport modules that handle intramonitor communication may be different from the transport modules that handle intermonitor communication. This allows the intramonitor transport modules to address security and reliability issues differently than how the intermonitor transport modules address security and reliability. While intramonitor communication may more commonly involve interprocess communication within a single host, intermonitor communication will most commonly involve cross-platform networked interoperation. For example, the intramonitor transport mechanisms may employ unnamed pipes which provides a kernel-enforced private interprocess communication channel between the monitor **16** components (this assumes a process hierarchy within the monitor **16** architecture). The monitor's **16** external transport, however, will more likely export data through untrusted network connections and thus require more extensive security management. To ensure the security and integrity of the message exchange, the external transport may employ public/private key authentication protocols and session key exchange. Using this same interface, third-party analysis tools may authenticate and exchange analysis results and configuration information in a well-defined, secure manner.

The pluggable transport permits flexibility in negotiating security features and protocol usage with third parties. Incorporation of a commercially available network management system can deliver monitoring results relating to security, reliability, availability, performance, and other

attributes. The network management system may in turn subscribe to monitor produced results in order to influence network reconfiguration.

All monitors (service, domain, and enterprise) 16a–16f use the same monitor code-base. However, monitors may include different resource objects 32 having different configuration data and methods. This reusable software architecture can reduce implementation and maintenance efforts. Customizing and dynamically configuring a monitor 16 thus becomes a question of building and/or modifying the resource object 32.

Referring to FIG. 3, the resource object 32 contains the operating parameters for each of the monitor's 16 components as well as the analysis semantics (e.g., the profiler engine's 22 measure and category definition, or the signature engine's 24 penetration rule-base) necessary to process an event stream. After defining a resource object 32 to implement a particular set of analyses on an event stream, the resource object 32 may be reused by other monitors 16 deployed to analyze equivalent event streams. For example, the resource object 32 for a domain's router may be reused as other monitors 16 are deployed for other routers in a domain 12a–12c. A library of resource objects 32 provides prefabricated resource objects 32 for commonly available network entities.

The resource object 32 provides a pluggable configuration module for tuning the generic monitor code-base to a specific event stream. The resource object 32 includes configurable event structures 34, analysis unit configuration 38a–38n, engine configuration 40a–40n, resolver configuration 42, decision unit configuration 44, subscription list data 46, and response methods 48.

Configurable event structures 34 define the structure of event records and analysis result records. The monitor code-base maintains no internal dependence on the content or format of any given event stream or the analysis results produced from analyzing the event stream. Rather, the resource object 32 provides a universally applicable syntax for specifying the structure of event records and analysis results. Event records are defined based on the contents of an event stream(s). Analysis result structures are used to package the findings produced by analysis engines. Event records and analysis results are defined similarly to allow the eventual hierarchical processing of analysis results as event records by subscriber monitors.

Event-collection methods 36 gather and parse event records for analysis engine processing. Processing by analysis engines is controlled by engine configuration 40a–40n variables and data structures that specify the operating configuration of a fielded monitor's analysis engine(s). The resource object 32 maintains a separate collection of operating parameters for each analysis engine instantiated in the monitor 16. Analysis unit configuration 38a–38n include configuration variables that define the semantics employed by the analysis engine to process the event stream.

The resolver configuration 42 includes operating parameters that specify the configuration of the resolver's internal modules. The decision unit configuration 44 describes semantics used by the resolver's decision unit for merging the analysis results from the various analysis engines. The semantics include the response criteria used to invoke countermeasure handlers. A resource object 32 may also include response methods 48. Response methods 48 include preprogrammed countermeasure methods that the resolver may invoke as event records are received. A response method 48 includes evaluation metrics for determining the circumstances under which the method should be invoked. These metrics include a threshold metric that corresponds to the measure values and scores produced by the profiler engine 22 and severity metrics that correspond to subsets of the associated attack sequences defined within the resource object 32.

Countermeasures range from very passive responses, such as report dissemination to other monitors 16a–16f or administrators, to highly aggressive actions, such as severing a communication channel or the reconfiguration of logging facilities within network components (e.g., routers, firewalls, network services, audit daemons). An active response may invoke handlers that validate the integrity of network services or other assets to ensure that privileged network services have not been subverted. Monitors 16a–16f may invoke probes in an attempt to gather as much counterintelligence about the source of suspicious traffic by using features such as traceroute or finger.

The resource object 32 may include a subscription list 46 that includes information necessary for establishing subscription-based communication sessions, which may include network address information and public keys used by the monitor to authenticate potential clients and servers. The subscription list 46 enables transmission or reception of messages that report malicious or anomalous activity between monitors. The most obvious examples where relationships are important involve interdependencies among network services that make local policy decisions. For example, the interdependencies between access checks performed during network file system mounting and the IP mapping of the DNS service. An unexpected mount monitored by the network file system service may be responded to differently if the DNS monitor informs the network file system monitor of suspicious updates to the mount requestor's DNS mapping.

The contents of the resource object 32 are defined and utilized during monitor 16 initialization. In addition, these fields may be modified by internal monitor 16 components, and by authorized external clients using the monitor's 16 API. Modifying the resource object 32 permits adaptive analysis of an event stream, however, it also introduces a potential stability problem if dynamic modifications are not tightly restricted to avoid cyclic modifications. To address this issue, monitors 16 can be configured to accept configuration requests from only higher-level monitors 16.

Referring to FIG. 4, a monitor performs network surveillance by monitoring 66 a stream of network packets. The monitor builds a statistical model of network activity from the network packets, for example, by building 68 long-term and short-term statistical profiles from measures derived from the network packets. The measures include measures that can show anomalous network activity characteristic of network intrusion such as measures that describe data transfers, network connections, privilege and network errors, and abnormal levels of network traffic. The monitor can compare 70 the long-term and short-term profiles to detect suspicious network activity. Based on this comparison, the monitor can respond 72 by reporting the activity to another monitor or by executing a countermeasure response. More information can be found in P. Porras and A. Valdes "Live Traffic Analysis of TCP/IP Gateways", Networks and Distributed Systems Security Symposium, March 1998, which is incorporated by reference in its entirety.

A few examples can illustrate this method of network surveillance. Network intrusion frequently causes large data

transfers, for example, when an intruder seeks to download sensitive files or replace system files with harmful substitutes. A statistical profile to detect anomalous data transfers might include a continuous measure of file transfer size, a categorical measure of the source or destination directory of the data transfer, and an intensity measure of commands corresponding to data transfers (e.g., commands that download data). These measures can detect a wide variety of data transfer techniques such as a large volume of small data transfers via e-mail or downloading large files en masse. The monitor may distinguish between network packets based on the time such packets were received by the network entity, permitting statistical analysis to distinguish between a normal data transfer during a workday and an abnormal data transfer on a weekend evening.

Attempted network intrusion may also produce anomalous levels of errors. For example, categorical and intensity measures derived from privilege errors may indicate attempts to access protected files, directories, or other network assets. Of course, privilege errors occur during normal network operation as users mistype commands or attempt to perform an operation unknowingly prohibited. By comparing the long-term and short-term statistical profiles, a monitor can distinguish between normal error levels and levels indicative of intrusion without burdening a network administrator with the task of arbitrarily setting an unvarying threshold. Other measures based on errors, such as codes describing why a network entity rejected a network packet enable a monitor to detect attempts to infiltrate a network with suspicious packets.

Attempted network intrusion can also be detected by measures derived from network connection information. For example, a measure may be formed from the correlation (e.g., a ratio or a difference) of the number of SYN connection request messages with the number of SYN_ACK connection acknowledgment messages and/or the number of ICMP messages sent. Generally, SYN requests received should balance with respect to the total of SYN_ACK and ICMP messages sent. That is, flow into and out-of a network entity should be conserved. An imbalance can indicate repeated unsuccessful attempts to connect with a system, perhaps corresponding to a methodical search for an entry point to a system. Alternatively, intensity measures of transport-layer connection requests, such as a volume analysis of SYN-RST messages, could indicate the occurrence of a SYN-attack against port availability or possibly port-scanning. Variants of this can include intensity measures of TCP/FIN messages, considered a more stealthy form of port scanning.

Many other measures can detect network intrusion. For example, "doorknob rattling," testing a variety of potentially valid commands to gain access (e.g., trying to access a "system" account with a password of "system"), can be detected by a variety of categorical measures. A categorical measure of commands included in network packets can identify an unusual short-term set of commands indicative of "doorknob-rattling." Similarly, a categorical measure of protocol requests may also detect an unlikely mix of such requests.

Measures of network packet volume can also help detect malicious traffic, such as traffic intended to cause service denials or perform intelligence gathering, where such traffic may not necessarily be violating filtering policies. A measure reflecting a sharp increase in the overall volume of discarded packets as well as a measure analyzing the disposition of the discarded packets can provide insight into unintentionally malformed packets resulting from poor line quality or internal errors in neighboring hosts. High volumes of discarded packets can also indicate more maliciously intended transmissions such as scanning of UPD ports or IP address scanning via ICMP echoes. Excessive number of mail expansion request commands (EXPN) may indicate intelligence gathering, for example, by spammers.

A long-term and short-term statistical profile can be generated for each event stream. Thus, different event streams can "slice" network packet data in different ways. For example, an event stream may select only network packets having a source address corresponding to a satellite office. Thus, a long-term and short-term profile will be generated for the particular satellite office. Thus, although a satellite office may have more privileges and should be expected to use more system resources than other external addresses, a profile of satellite office use can detect "address spoofing" (i.e., modifying packet information to have a source address of the satellite office).

The same network packet event may produce records in more than one event stream. For example, one event stream may monitor packets for FTP commands while another event stream monitors packets from a particular address. In this case, an FTP command from the address would produce an event record in each stream.

Referring to FIG. 5, a monitor may also "deinterleave." That is, the monitor may create and update 74, 76 more than one short-term profile for comparison 78 against a single long-term profile by identifying one of the multiple short-term profiles that will be updated by an event record in an event stream. For example, at any one time a network entity may handle several FTP "anonymous" sessions. If each network packet for all anonymous sessions were placed in a single short-term statistical profile, potentially intrusive activity of one anonymous session may be statistically ameliorated by non-intrusive sessions. By creating and updating short-term statistical profiles for each anonymous session, each anonymous session can be compared against the long-term profile of a normal FTP anonymous session. Deinterleaving can be done for a variety of sessions including HTTP sessions (e.g., a short-term profile for each browser session).

Referring to FIG. 6, a computer platform 14 suitable for executing a network monitor 16 includes a display 50, a keyboard 54, a pointing device 58 such as a mouse, and a digital computer 56. The digital computer 56 includes memory 62, a processor 60, a mass storage device 64*a*, and other customary components such as a memory bus and peripheral bus. The platform 14 may further include a network connection 52.

Mass storage device 64*a* can store instructions that form a monitor 16. The instructions may be transferred to memory 62 and processor 60 in the course of operation. The instructions 16 can cause the display 50 to display images via an interface such as a graphical user interface. Of course, instructions may be stored on a variety of mass storage devices such as a floppy disk 64*b*, CD-ROM 64*c*, or PROM (not shown).

Other embodiments are within the scope of the following claims.

What is claimed is:

1. A method of network surveillance, comprising:

receiving network packets handled by a network entity;

building at least one long-term and at least one short-term statistical profile from at least one measure of the network packets, the at least one measure monitoring data transfers, errors, or network connections;

comparing at least one long-term and at least one short-term statistical profile; and

determining whether the difference between the short-term statistical profile and the long-term statistical profile indicates suspicious network activity.

2. The method of claim 1, wherein the measure monitors data transfers by monitoring network packet data transfer commands.

3. The method of claim 1, wherein the measure monitors data transfers by monitoring network packet data transfer errors.

4. The method of claim 1, wherein the measure monitors data transfers by monitoring network packet data transfer volume.

5. The method of claim 1, wherein the measure monitors network connections by monitoring network connection requests.

6. The method of claim 1, wherein the measure monitors network connections by monitoring network connection denials.

7. The method of claim 1, wherein the measure monitors network connections by monitoring a correlation of network connections requests and network connection denials.

8. The method of claim 1, wherein the measure monitors errors by monitoring error codes included in a network packet.

9. The method of claim 8, wherein an error code comprises a privilege error code.

10. The method of claim 8, wherein an error code comprises an error code indicating a reason a packet was rejected.

11. The method of claim 1, further comprising responding based on the determining whether the difference between the short-term statistical profile and the long-term statistical profile indicates suspicious network activity.

12. The method of claim 11, wherein responding comprises transmitting an event record to a network monitor.

13. The method of claim 12, wherein transmitting the event record to a network monitor comprises transmitting the event record to a hierarchically higher network monitor.

14. The method of claim 13, wherein transmitting the event record to a network monitor comprises transmitting the event record to a network monitor that receives event records from multiple network monitors.

15. The method of claim 14, wherein the monitor that receives event records from multiple network monitors comprises a network monitor that correlates activity in the multiple network monitors based on the received event records.

16. The method of claim 11, wherein responding comprises altering analysis of the network packets.

17. The method of claim 11, wherein responding comprises severing a communication channel.

18. The method of claim 1, wherein the network packets comprise TCP/IP packets.

19. The method of claim 1, wherein the network entity comprises a gateway, a router, or a proxy server.

20. The method of claim 1, wherein the network entity comprises a virtual private network entity.

21. A method of network surveillance, comprising:

monitoring network packets handled by a network entity;

building a long-term and multiple short-term statistical profiles of the network packets;

comparing one of the multiple short-term statistical profiles with the long-term statistical profile; and

determining whether the difference between the one of the multiple short-term statistical profiles and the long-term statistical profile indicates suspicious network activity.

22. The method of claim 21, wherein the multiple short-term statistical profiles comprise profiles that monitor different anonymous FTP sessions.

23. The method of claim 21, wherein building multiple short-term statistical profiles comprises deinterleaving packets to identify a short-term statistical profile.

24. A computer program product, disposed on a computer readable medium, the product including instructions for causing a processor to:

receive network packets handled by a network entity;

build at least one long-term and at least one short-term statistical profile from at least one measure of the network packets, the measure monitoring data transfers, errors, or network connections;

compare at least one short-term and at least one long-term statistical profile; and

determine whether the difference between the short-term statistical profile and the long-term statistical profile indicates suspicious network activity.

25. A method of network surveillance, comprising:

receiving packets at a virtual private network entity; and

building at least one long-term and at least one short-term statistical profile based on the received packets, and

comparing at least one long-term statistical profile with at least one short-term statistical profile to determine whether the packets indicate suspicious network activity.

26. The method of claim 25, further comprising decrypting the packets before statistically analyzing the packets.

27. The method of claim 25, further comprising not decrypting the packets before statistically analyzing the packets.

* * * * *

G

Technical Report
(CDRL A005)

Architecture Design of a Scalable Intrusion Detection System for the Emerging Network Infrastructure

DARPA Order Number: E296

Issued by Rome Lab under Contract Number: F30602-96-C0325

Submitted: April 1997

by

Y. Frank Jou
Fengmin Gong
Chandru Sargor

Shyhtsun Felix Wu
W. Rance Cleaveland

MCNC
Information Technologies Division
Research Triangle Park, N.C. 27709

Dept. of Computer Science
North Carolina State University
Raleigh, N.C. 27695-7914

Contents

List of Tables

List of Figures

Design of a Scalable Intrusion Detection System for the Emerging Network Infrastructure

1 Introduction

This document describes the design of a scalable intrusion detection system funded by DARPA through Contract No. F30602-96-C0325. This three-year project aims at designing and developing a software system for protecting against intruders from breaking into network routers, switches, and network management channels. The project is a joint collaboration between MCNC and North Carolina State University (NCSU).

Given the increasing popularity of the Internet, intrusion incidents are becoming common events of life. Some of these incidents are simply out of innocuous curiosity. Some, however, are due to malicious attempts in order to compromise the availability of information system or the integrity and privacy of the information itself. Despite the best efforts of the protocol designers, implementors, and system administrators, it is prudent to assume that attacks will occur and some, unfortunately, will succeed. Therefore, it is vitally important to develop means to automatically detect and respond to these attacks in order to maintain critical information services.

Depending on the goals of the intruder, the targets of attack may range from individual end hosts to a network of routers and switches. In this project, we focus our effort on the protection of the network infrastructure since the attacks on the routers/switches have the potential of disrupting a large scale of information services on which the national defense and economy may depend. Our goal of designing this detection system is to provide a comprehensive approach which leverages on the application of novel detection techniques together with extension of some existing host-based intrusion detection methods in an internetworking environment. In particular, we will conduct logical and statistical analysis of network routing and management protocols to construct a scalable distributed intrusion detection system for the emerging internetwork environment.

1.1 Background

Intrusive activity is occurring on our computer systems. Reports frequently appear in the media about outsiders breaking into computers, employees misusing computers, and rogue viruses and worms penetrating computer systems. Due to these incidents, we have seen a growing interest in computer system intrusion detection in the last several years. The earliest work in this area was a study by Jim Anderson [1]. Anderson categorized the threats as:

- Masquerader: An individual who is not authorized to use the computer, and who penetrates a system's access controls to exploit a legitimate user's account.
- Misfeasor: A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges.

- Clandestine users: An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection altogether.

Anderson suggested that masqueraders can be detected either by auditing failed login attempts or by observing departures from established patterns of use for individual users. Misfeasors can be detected by observing failed access attempts to files, programs, and other resources. His suggestion for detecting the clandestine user is to monitor certain system-wide parameters, such as CPU, memory, and disk activity, and compare these with what has been historically established as "usual" or "normal" for that facility. All of these approaches have been adopted one way or the other by subsequent studies.

Dorothy Denning [4] and her colleagues at SRI International undertook a project for developing an intrusion detection expert system (IDES) prototype. Denning proposed to monitor standard operations on a target system for deviations in usage. Her early research tried to define the activities and statistical measures best suited to do this detection. Teresa Lunt [5] and her colleagues continue this research with the development of the IDES system. They expanded the original concept by adding an expert system component that addresses known or suspected security flaws in the target system. IDES (and its follow-up Next generation IDES, or NIDES) system research has served to demonstrate two things. First, statistical analysis of computer system activities provides a characterization of normal system and user behavior, and activities deviating beyond normal bounds is detectable. Second, known intrusion scenarios, exploitation of known system vulnerabilities, and violations of a system's security policy are detectable through use of a rule-based expert system.

In the early stage, intrusion detection system were designed around the analysis of a single host's audit trail. With the proliferation of computer networks, many of the intrusion detection systems began to extend the techniques to networks of computers. Most of the current network intrusion detection efforts have taken one of the two following approaches. One approach is to collect data from separate hosts on a network for processing by a centralized intrusion detection system [2][3]. The other approach is to target network traffic at the service and protocol levels [6][7]. Our effort is close to the second approach with a few exceptions. First, we are interested in protecting network infrastructure and particularly focus on routing and management capabilities. Therefore, the target of analysis is mainly on specific protocol traffic instead of general data traffic. Second, the proposed protocol analysis approach in our architecture design is unique which analyzes the logical behavior of routing and management protocols in order to identify the set of states that are indicative of security attacks. Third, network management functionalities are part of the integrated system design. Through these functionalities, the intrusion detection system can be incorporated into existing management framework as an extension of fault management.

1.2 Organization

Section 2 provides an overview of the architecture of a model intrusion detection system and introduce its components and associated functional requirements. Section 3 outlines the design objectives and system features for the experimental system being implemented by the authors. Finally, detailed description of a functional overview is given in Section 4.

2 Intrusion Detection System Architecture

In this section we present an overview of the system architecture design. The system consists of complementary functional blocks for providing comprehensive detection capabilities. It also incorporates standard network management functionalities to lay a foundation for facilitating automated responses in future research efforts. A brief description of each system component and its functional requirements will be given later in the section.

2.1 Architecture Overview

Figure 1 illustrates the architecture design of our intrusion detection system. At the top level, there are two subsystems: namely, local detection subsystem and remote management subsystem. The remote management unit implements a set of network management applications which can both probe the status of and issue commands to the local detection subsystem. It is one of our design objectives that the management applications will be based on SNMP such that the management function can be easily incorporated into any existing SNMP based network management platform.

A local subsystem is associated with a router/switch to function as a security filter and analyze the incoming packets from its neighbors. The transaction record with each neighbor will be maintained separately. If any of its neighbor routers/switches behaves differently from its historical norm or transitions into an improper protocol state, then it may be an indication that this neighbor is either faulty or compromised. Depending on the degree of deviation or the nature of fault/attack, an alert or alarm signal will be issued to acquire the security officer's attention.

A remote management subsystem can oversee several routers/switches. Some intrusions, like doorknob rattling attack, which may be difficult to detect at a local level can be made easier by checking the global status across several routers/switches. While it is not within the scope of this project, we expect that the detection/analysis functions implemented in the local subsystem can be extended to a global level and correlate intrusion events among several routers. The management capability, which is based on SNMP framework, can logically be further extended among management nodes in a hierarchical fashion to establish a status map for an autonomous system.

2.2 Components

The functional requirements of the components shown in Figure 1 are described in the following sections.

2.2.1 Local Subsystem

A local subsystem consists of the following modules: interception/redirection module, rule-based prevention module, protocol and statistical-based detection modules, decision module, and information abstraction module. It also includes a management information base (MIB) and remote MIB agent functions which provide access to remote management applications. A brief description for each module follows.

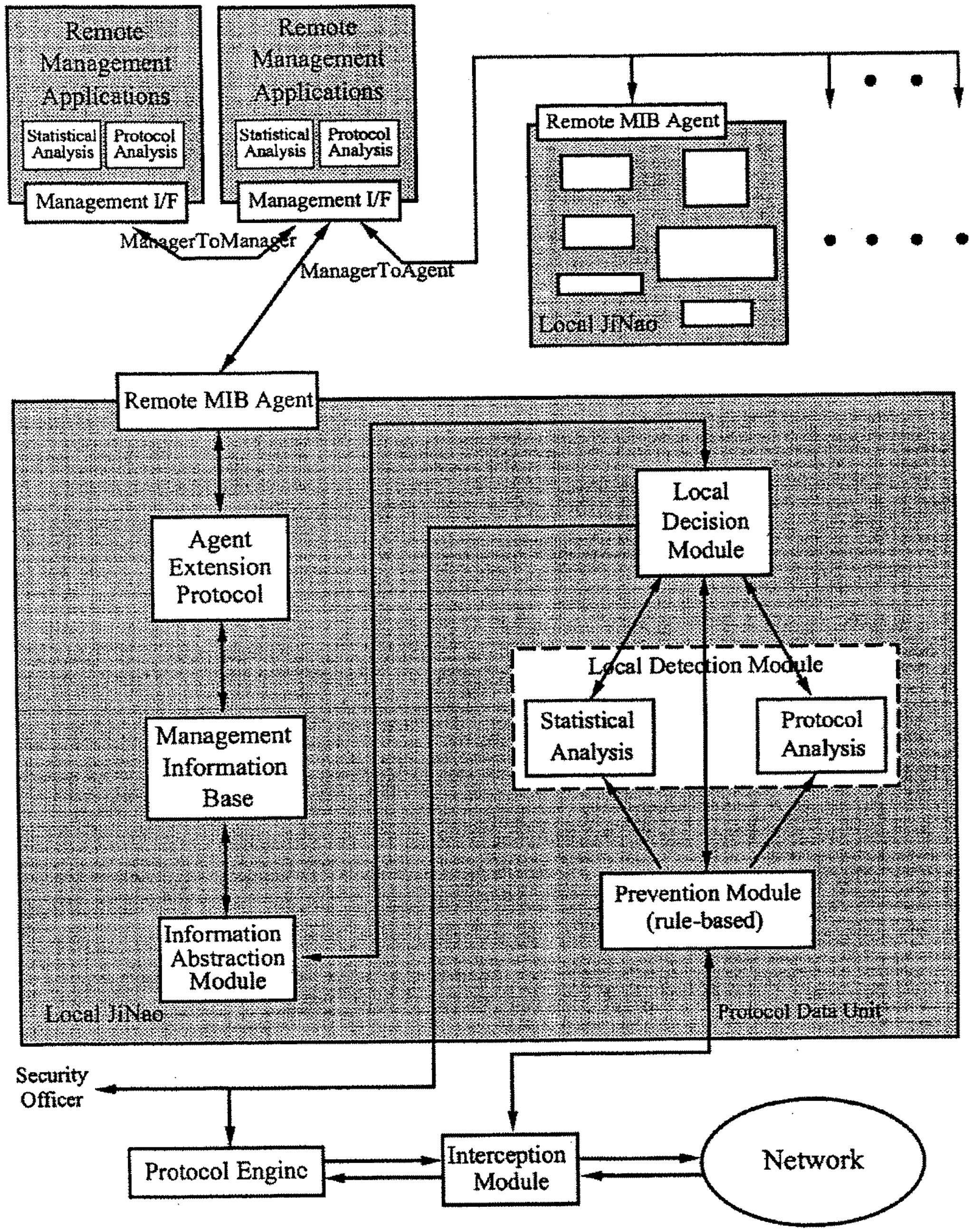


Figure 1: Ji-Nao System Architecture.